Exercise 2 - Structure Prediction for PoS Tagging

Lirane Bitton 200024677

24/11/2018

Part I

Theoretical Questions

1 MEMM "contains" HMM

Let $e(\cdot,\cdot)$ and $t(\cdot,\cdot)$ be the emission and transission probabilities respectively of ou HMM model. Furthermore let assume that the special START_STATE tag is part of our tags (hence y_0 is START_STATE with probability 1).

$$\begin{aligned}\mathbb{P}[Y_{1:n}, X_{1:n}]_{HMM} &= \prod_t \mathbb{P}[y_t|y_{t-1}] \cdot \mathbb{P}[x_t|y_t] \\ &= \prod_t t(y_{t-1}, y_t) \cdot e(y_t, x_t)\end{aligned}$$

For the MEMM model:

$$\mathbb{P}[Y_{1:n}, X_{1:n}]_{MEMM} = \prod_t \frac{e^{\phi(y_t, y_{t-1}, X_{1:n}, t)\cdot w^T}}{\sum_k e^{\phi(k, y_{t-1}, X_{1:n}, t)\cdot w^T}}$$

Let define w to be of length $|pos|^2 + |pos| \cdot |words|$ where the $|pos|^2$ indices corresponds to the transmissions between states, i.e., $w(i,j) = \log(t(i,j))$ in the same way for the emission:

$$\begin{aligned}
\mathbb{P}[Y_{1:n}, X_{1:n}]_{MEMM} &= \prod_t \frac{e^{\phi(y_t, y_{t-1}, X_{1:n}, t)\cdot w^T}}{\sum_k e^{\phi(k, y_{t-1}, X_{1:n}, t)\cdot w^T}} \\
&= \prod_t \frac{e^{\log(t(y_{t-1}, y_t)) + \log(e(y_t, x_t)) + \log[\sum_k e^{\phi(k, y_{t-1}, X_{1:n}, t)\cdot w^T}]}}{\sum_k e^{\phi(k, y_{t-1}, X_{1:n}, t)\cdot w^T}} \\
&= \prod_t \frac{t(y_{t-1}, y_t) \cdot e(y_t, x_t) \cdot \sum e^{\phi(k, y_{t-1}, X_{1:n}, t)\cdot w^T}}{\sum_k e^{\phi(k, y_{t-1}, X_{1:n}, t)\cdot w^T}} \\
&= \prod_t t(y_{t-1}, y_t) \cdot e(y_t, x_t) \\
&= \mathbb{P}[Y_{1:n}, X_{1:n}]_{HMM} \blacksquare
\end{aligned}$$

2 Higher Order Markov Model

A second-order markov model can be written as a log linear model . Actually conditioning on other variables means that we need to make our w and ϕvectors longer, in the case describe in the question we will build them as stated above except that now the transmission part of the vector will comprise $|pos|^3$parameters and the total vector $|pos|^3 + |pos| \cdot |words|$.

3 Energy Based Model Gradient

a. Then the energy function is $E\left(y_t | y_{t-1}, x_{1:T}; \theta\right) = -w^T \phi\left(y_{i-1}, y_i, x_{1:n}\right)$ since:

$$\begin{aligned}
\mathbb{P}[y_t | y_{t-1}, x_{1:T}] &= \frac{1}{\sum_{y_t} e^{-E(y_t | y_{t-1}, x_{1:T}; \theta)}} e^{-E(y_t | y_{t-1}, x_{1:T}; \theta)} \\
&= \frac{1}{\sum_{y_t} e^{w^T \phi(y_{i-1}, y_i, x_{1:n})}} e^{w^T \phi(y_{i-1}, y_i, x_{1:n})} \\
&= \mathbb{P}[y_t | y_{t-1}, x_{1:n}]_{MEMM}
\end{aligned}$$

b.

$$\mathbb{E}_{x \sim D}[\frac{\partial \log \mathbb{P}_\theta(x)}{\partial \theta}] = \sum_{x \sim D} \mathbb{P}(x) \cdot [\frac{\partial \log \mathbb{P}_\theta(x)}{\partial \theta}] \quad *$$

denote that:

$$\log \mathbb{P}_\theta(x) = \log \frac{e^{-\mathbb{E}(x;\theta)}}{\sum_x e^{-\mathbb{E}(x;\theta)}} = -\mathbb{E}(x;\theta) - \log(\sum_x e^{-\mathbb{E}(x;\theta)})$$

let derive $\log(\sum_x e^{-\mathbb{E}(x;\theta)})$:

$$\frac{\log(\sum_x e^{-\mathbb{E}(x;\theta)})}{\partial\theta} = \sum_x \left[\frac{e^{-\mathbb{E}(x;\theta)}}{\sum_x e^{-\mathbb{E}(x;\theta)}} \cdot \frac{\partial(-\mathbb{E}(x;\theta))}{\partial\theta}\right] = \sum_x \mathbb{P}(x) \cdot \frac{\partial(-\mathbb{E}(x;\theta))}{\partial\theta} = \mathbb{E}[\frac{\partial(-\mathbb{E}(x;\theta))}{\partial\theta}]$$

Let assign in *:

$$\begin{aligned}
&\sum_{x\sim D} \mathbb{P}(x) \cdot \left[[\frac{\partial(-\mathbb{E}(x;\theta))}{\partial\theta}] - \sum_{x\sim D} \mathbb{E}_{x\sim\theta}[\frac{\partial(-\mathbb{E}(x;\theta))}{\partial\theta}]\right] \\
&= \mathbb{E}_{x\sim D}[\frac{\partial(-\mathbb{E}(x;\theta))}{\partial\theta}] - \sum_{x\sim D} \mathbb{P}(x) \cdot \mathbb{E}_{x\sim\theta}[\frac{\partial(-\mathbb{E}(x;\theta))}{\partial\theta}] \\
&= \mathbb{E}_{x\sim D}[\frac{\partial(-\mathbb{E}(x;\theta))}{\partial\theta}] - \mathbb{E}_{x\sim\mathbb{P}_\theta}[\frac{\partial(-\mathbb{E}(x;\theta))}{\partial\theta}] \\
&= \mathbb{E}_{x\sim\mathbb{P}_\theta}[\frac{\partial(\mathbb{E}(x;\theta))}{\partial\theta}] - \mathbb{E}_{x\sim D}[\frac{\partial(\mathbb{E}(x;\theta))}{\partial\theta}]
\end{aligned}$$

4 MLE for HMM

Let derive the MLE seen in class for $t_{i,j}$

$$\ell(S,\theta) = \sum_{i=1}^{N}\sum_{t=1}^{T_i} \log\left(t_{y_{t-1},y_t}\right) + \log\left(e_{y_t,x_t}\right)$$

subject to

$$\forall i : \sum_j t_{i,j} = 1$$
$$\forall i,j : t_{i,j} \geq 0$$

Using the Lagrangian $L(S,\theta) = \ell(S,\theta) + \sum_i \lambda_i (1 - \sum_k t_{i,k})$.
We look at the partial derivative w.r.t $t_{i,j}$ and compare it to 0:

$$\begin{aligned}
&\frac{\partial L}{\partial t_{i,j}} = \frac{1}{t_{i,j}} \cdot \#(i \to j) - \lambda_i = 0 \\
&\implies t_{i,j} = \frac{\#(i \to j)}{\lambda_i}
\end{aligned}$$

Now we look at the partial derivative w.r.t λ_i:

$$\begin{aligned}
&\frac{\partial L}{\partial \lambda_i} = \left(1 - \sum_k t_{i,k}\right) = 0 \\
&\implies 1 - \sum_k \frac{\#(i \to k)}{\lambda_i} = 0 \\
&\implies \lambda_i = \sum_k \#(i \to k)
\end{aligned}$$

Then $t_{i,j}$ is with λ as a normalization parameter:

$$t_{i,j} = \frac{\#(i \to j)}{\lambda_i} = \frac{\#(i \to j)}{\sum_k \#(i \to k)}$$

Part II
Practical Exercise

4.1 Usage

The program can be run without parameter and you'll get the results of the HMM and Baseline tests on 10% of the data while trained on 10, 25 and 90 percent of the data plus printing 5 sentences sampled by the HMM learned on 90%. Otherwise you can run the program with parameters and your script should be

```
python3 "PoS Tagging.py" percent_for_training percent_for_testing eta epochs file_to
```

where percent for training is the first percent of the data and percent for testing is from that percent of data on.

4.2 Implementation

The implementation of the model was done as learned in class while there are some preprocessing for the MEMM like hold a dict of all transission/emission in order to reduce runtime. The MEMM was implemented with the feature of HMM.

About the data I added the start and end tags and words to each sequence.

4.3 Results & Discussion

Here are the results for the training set size you recommended. For a while I had some problem with my MEMM learning phase. Actually it wasn't leaning at all or almost all (~10% success). Hence I decided to fit my data on a small piece of data (~0.25% of the data) and train it and test on it, also because of runnning time exhaustive. That the way I found to find the bug I had (which actually was due to a copy past where I forgot to put a minus before eta on the predicted results). Then I began to play with the different parameter and test my small data learned model on some other pieces of the data set. With one epoch tested on the data I learned from I got 73% success then with two and three epochs I got 85% and 88% but when testing on some other random small dataset I got accuracies varying from 54% to 61% the best one were about 60% with epochs. Playing with eta parameters somehow give some better results between 60 to 66% until a learning rate of 0.3. After that value each time I

	10% training	25% training	90% training
Baseline	83.109%	87.235%	91.022%
HMM	89.026%	91.869%	94.5864%
MEMM(HMMFeature)	83.386%	87.177%	90.840%
MEMM 2 epochs	86.669%		
MEMM 3 epochs	87.103%		

Table 1: Test result on 10% of the dataset

increased I got some worse results except while I tested on the learned dataset. Hence for some eta value (depending on dataset size) there is a risk of fitting the trainig data in place of learning. Then here again I'm sure that if I had enough time to tune the eta value I could have been got some better results, perhaps better that the HMM.

The model were train and checked on the 10 last percent of the dataset provided with default parameter for the MEMM perceptron learning (1 epoch, zero initialization and eta=0.1)

4.4 Sample sentence HMM

['The', 'researchers', "'m", 'about', 'negotiating', 'to', 'save', 'its', 'delivery' 'beneficial', 'economies', 'of', 'a', 'dollars', 'after', 'him', 'compares', 'limited', at', 'net', 'sales']

Acually it sounds like english, I think it can generate and idea but not a correct sentence.